

48K
Spectrum
Nº 11
sinclair
AÑO I
REVISTA MENSUAL COLECCIONABLE
P.V.P.
395
IVA Inc.
ANALISIS SOFT
NIGHTSHADE
CURSO CODIGO MAQUINA X PARTE
Juego del mes
CAMARA SECRETA


TYPE AND RUN
LA PRIMERA REVISTA CON CASSETTE PARA TECLEAR Y
AÑO I – N.º 8
LISTADOS PARA
SPECTRUM
ANUALIDADES
SALON
COMMODORE
BUSQUEDA
FICHAS
CIRCUITO
MENSUAL
Pídala en su kiosko
o en tiendas especializadas
SUSCRIPCIONES: MONSER, S.A. C/ Argos, 9 - 28037 MADRID
510 CASSETTE
ESPECIAL PARA
ORDENADOR
MONSER
C-10
LA MEJOR REVISTA CON CASSETTE
Monser



Sumario:

AÑO I - N.º 11 - 1986

DIRECTOR
José Nieto Rubio

COORDINADOR
Félix Santamaría Avila

SUPERVISOR SOFTWARE
Gustavo Cano Muñoz

DISEÑO
Marco Antonio Díaz

REDACCION Y COLABORADORES
Nieves Márquez
Victoria Aguilar
Javier González

PORTADA
Mauro Novoa

EDITA
MONSER, S. A.

DIRECTOR EDITORIAL
J. L. Cano Regidor

REDACCION, ADMINISTRACION Y PUBLICIDAD
Argos, 9
28037-MADRID
Tel. 742 72 12/96

PUBLICIDAD Y SUSCRIPTORES
Esther Martínez

FOTOCOMPOSICION
CRISOL, S. A.
Virgen del Val, 48

FOTOMECANICA
IMAGEN

IMPRIME
Gráficas Marte, S. A.

DISTRIBUCION
R.B.A.
Barranco de la Presa, 23
Fuenlabrada -MADRID-

Depósito legal: M-10.328-1985

Se solicitará control O.J.D.

Pasatiempo

Es conocido de antiguo que la ociosidad es sumamente perniciosa. Hay quien no duda ni un instante en señalarla con el dedo de acusar como la verdadera madre de todos los vicios.

Ante esto, nosotros, naturalmente, no podíamos permanecer impasibles y hemos interrumpido nuestra ociosidad para ponernos delante del ordenador y construir un programa que interrumpa la vuestra.

Y lo hemos hecho de tal manera que incluso esas pequeñas ociosidades que a ojos bienintencionados pudieran parecer inofensivas, esos ratitos de tiempo sin nada que hacer, se transformen en un edificante y beneficioso galopar de las neuronas, en un desenfrenado ímpetu matemático, en un traer y llevar de numeritos más propio de un contable loco que de un semicuerdo aficionado a los ordenadores.

En otras palabras, ¡hemos hecho un pasatiempo!

Convencidos como estamos de que vuestro arrollador ímpetu os va a llevar, si cabe, aún más lejos y de que vuestra hambre programadora no os permitirá descansar hasta añadir al programa esos detalles personales que tanto os gustan, os describiremos más adelante detalladamente lo más sustancioso del programa.

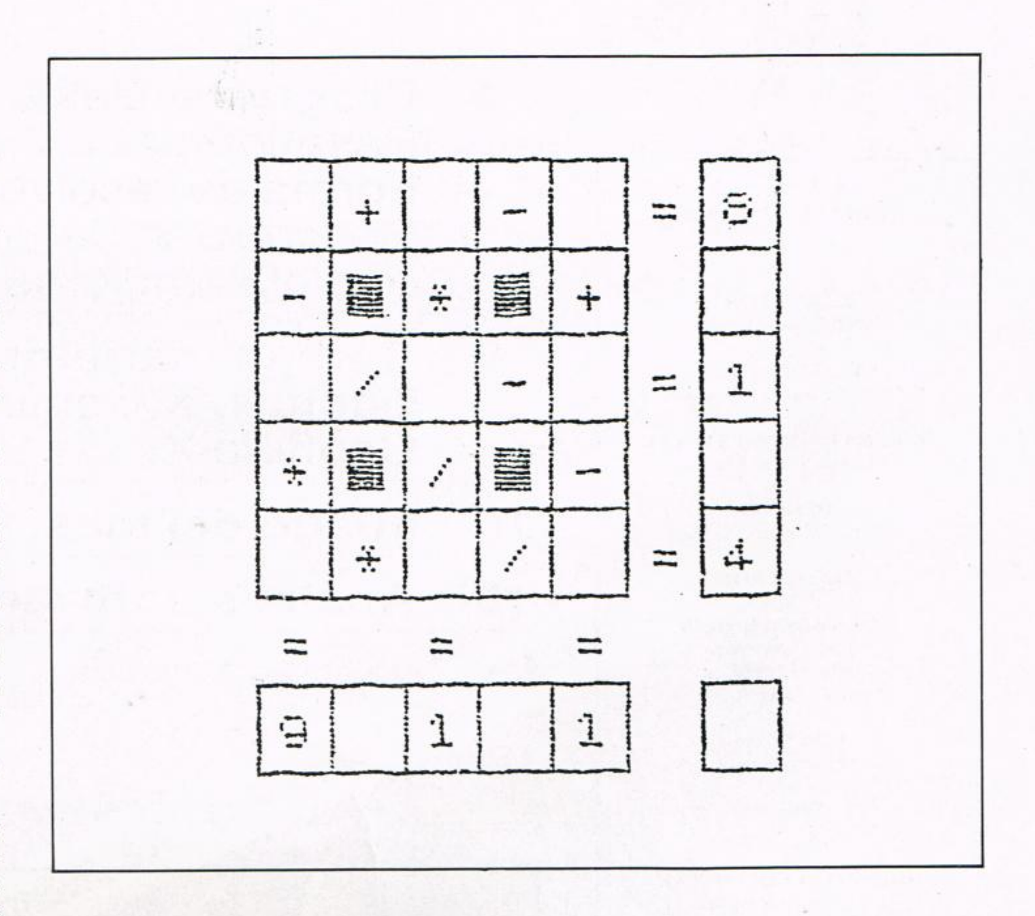
Forma de resolverlo

Debéis encontrar números enteros positivos comprendidos entre 1 y 9, ambos inclusive, tales que al realizar las operaciones indicadas en el recuadro, tanto horizontal como verticalmente, den los resultados que también se indican en la figura. Es importante tener en cuenta que la segunda operación debe realizarse sobre el resultado de la primera, de tal forma que puede suponerse un paréntesis incluyendo las dos primeras cifras y el signo que va en medio. Veamos un ejemplo:

$$3 + 4 \times 5$$

debe ser considerado equivalente a

$$(3 + 4) \times 5 = 7 \times 5 = 35$$



Descripción del programa

Lineas 1 a 40

Se ejecutan sólo la primera vez que se carga el programa. Contienen la definición y el dimensionado de variables A$ (5,5) contendrá las 5 filas y columnas de que consta el pasatiempo, es decir, tanto las filas y columnas que contienen cifras como las que no las contienen. La línea 40 bifurca a obtener un primer juego antes de presentarlo en pantalla conjuntamente con las opciones disponibles en ese momento.

Líneas 50 a 180

Son las líneas que contienen la presentación de opciones y la distribución del control a la opción solicitada. Mientras no se pulse ninguna tecla válida, el programa se estará ejecutando entre las instrucciones 100 y 120. Cuando se pulsa un 1, un 2 ó un 3, el control se transfiere a las líneas 500, 600 ó 700 respectivamente. El control retornará finalmente a la instrucción 50, cuando la opción seleccionada haya sido ejecutada.

Líneas 500 a 599

El control llega a ellas con la opción 1, es decir, cuando se desea obtener un nuevo pasatiempo. Este puede ser un lugar adecuado para añadir cosas al programa. Se hace una llamada a una rutina, situada en la línea 1000, que es la que verdaderamente obtiene un nuevo pasatiempo.

Líneas 600 a 699

Corresponden a la opción 2, que dibuja el pasatiempo sin generar uno nuevo. Esta opción, es especialmente útil si se introducen modificaciones que alteren la pantalla y exijan redibujar el pasatiempo. Funciona mediante llamada a la subrutina situada en la línea 3000.

MA BASIC

Líneas 700 a 799

Se encarga de imprimir la solución del pasatiempo. Llama a la subrutina situada en la línea 3300.

Línea 1000

Esta rutina es el verdadero corazón del programa. Llama a su vez a dos subrutinas, que son las más complejas del programa (las de las líneas 5000 y 7000).

Cuando las filas y columnas que se van generando son tales que sea imposible encontrar una solución se recalculan tantas columnas como sea necesario (líneas 1400 a 1440).

Líneas 3000 a 3070

Sirven para dibujar el pasatiempo sin incluir la solución. Utilizan la misma programación que el dibujo que contiene la solución. Poniendo n$ = «n» consigue que la solución no sea dibujada.

Líneas 3300 a 3999

Hacen n$ = «» para indicar que se debe dibujar el pasatiempo con su solución.

Líneas 5000 a 5900

Esta subrutina genera de forma pseudo-aleatoria, una línea horizontal (fila) de las 3 que hay conteniendo números y signos y se almacena en z$. Esta rutina es llamada desde la situada en la línea 1000, que moverá los valores de z$ a la matriz a$ (filas 1, 3 ó 5). Números y signos se generan de tal forma que cumplan la condición de que los resultados intermedios sean números enteros positivos y que el resultado final también lo sea, así como que esté comprendido entre 1 y 9 (ambos inclusive).

Se tiene especial cuidado en evitar errores que siempre se pueden producir al no poder trabajar en el SPECTRUM con variables definidas como enteras. Esto obliga a utilizar con preferencia la función INT.

En algunos casos se hace necesario volver a empezar la operación (véase por ejemplo la instrucción 5390).

Líneas 7000 a 7360

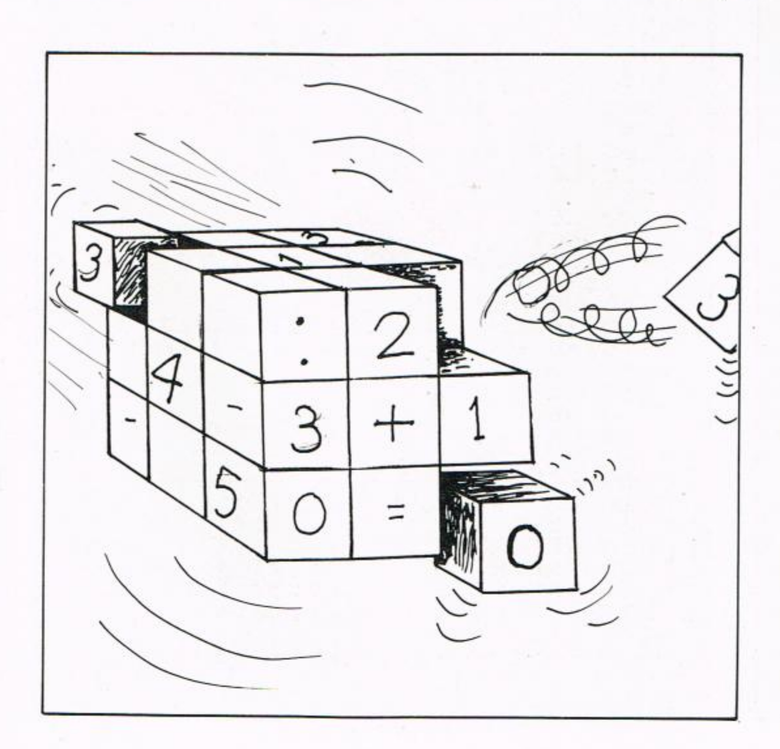
Se llaman desde la rutina 1000. Aquí se calculan los signos de las columnas 1, 3 y 5 de a$, ya que los números ya fueron calculados al obtener las filas. También aquí se utiliza z$ como variable intermedia. Sus valores se acabarán moviendo a las columnas 1, 3 ó 5 de a$.

Se van tanteando pasos de signos. La pareja por la que se empieza a tantear se obtiene aleatoriamente. Si no se encontrase ninguna válida, se devuelve control con una «i» en r$ a fin de volver a calcular alguna o todas las filas obtenidas anteriormente.

Posibles mejoras a introducir

Si tenéis impresora, podeis introducir la posibilidad de dar un COPY de pantalla ya que puede ser muy útil disponer de varias copias en papel que permitan probar números sin emborronar mucho. Para los más atrevidos se me ocurre la idea de ir escribiendo los números que se están probando sobre la pantalla.

Otras posibilidades fácilmente añadibles al programa serían que diese pistas, etc.




```
   1 REM PASATIEMPO
   2 REM EJECUCION [illegible]
   5 BORDER 7: INK 0: PAPER 7: B
RIGHT 0: CLS
  10 DIM a$(5,5)
  20 DIM z$(5)
  30 DIM s$(4)
  32 LET s$(1)="+"
  34 LET s$(2)="-"
  36 LET s$(3)="*"
  38 LET s$(4)="/"
  40 GO TO 500
  50 REM bucle permanente
  51 REM opciones
  55 PRINT AT 21,0;"Opciones:"
  70 PRINT AT 21,10;"1-Nuevo jue
go"
  80 PRINT #1;AT 0,10;"2-Repetir
 juego"
  90 PRINT #1;AT 1,10;"3-Imprimi
r solucion"
 100 LET x=CODE INKEY$-48
 120 IF x<1 OR x>3 THEN GO TO 10
0
 140 IF x=1 THEN GO TO 500
 160 IF x=2 THEN GO TO 600
```

```
 180 GO TO 700
 500 REM OPCION 1.-Obtener pasat
iempo
 520 PRINT AT 9,10; FLASH 1;"CAL
CULANDO"
 550 GO SUB 1000
 600 REM OPCION 2.-Dibujar pasat
iempo
 610 GO SUB 3000
 699 GO TO 50
 700 REM OPCION 3.-Dibujar soluc
ion
 710 GO SUB 3300
 799 GO TO 50
 999 REM RUTINAS
1000 REM Rutina de obtencion de
filas
1005 REM 1a fila
1010 LET imp=0
1020 GO SUB 5000
1030 LET a$(1)=z$
1100 REM 3a fila
1120 GO SUB 5000
1130 LET a$(3)=z$
1200 REM 5a fila
1220 GO SUB 5000
1230 LET a$(5)=z$
1300 REM obtencion de columnas
1310 FOR c=1 TO 5 STEP 2
1320 LET r$=" "
1330 GO SUB 7000
1340 IF r$="i" THEN GO TO 1400
1380 NEXT c
1390 RETURN
1400 REM Recalcular columnas
1410 LET imp=imp+1
1420 IF imp=1 THEN GO TO 1200
1430 IF imp=2 THEN GO TO 1100
1440 GO TO 1000
3000 REM Rut.dibujar pasatiempo
3020 LET n$="n"
3050 CLS
3070 GO TO 3350
3300 REM Rut.dibujar pasatiempo
o solucion
3340 LET n$="s"
3350 FOR x=1 TO 8
3380 PLOT 60,172-16*x: DRAW 80,0
: PLOT 156,172-16*x: DRAW 16,0
3400 PLOT 44+16*x,156: DRAW 0,-8
0: PLOT 44+16*x,60: DRAW 0,-16
3420 NEXT x
3440 FOR x=5 TO 9 STEP 4
3460 FOR y=10 TO 14 STEP 4
3470 PRINT AT x,y;"█"
3480 NEXT y
3490 NEXT x
3500 FOR x=1 TO 5
3510 FOR y=1 TO 5
3520 IF n$="n" AND 2*INT (x/2+.5
)<>x AND 2*INT (y/2+.5)<>y THEN
GO TO 3535
3530 IF a$(x,y)<>" " THEN PRINT
AT 2*x+1,2*y+6;a$(x,y)
3535 NEXT y
3540 IF x=2 OR x=4 THEN GO TO 35
60
3550 PRINT AT 2*x+1,18;"=";AT 2*
x+1,20;VAL ("("+a$(x,1 TO 3)+")"
+a$(x,4 TO ))
3560 NEXT x
3570 PRINT AT 13,8;"=";AT 13,12;
"=";AT 13,16;"="
3580 FOR c=1 TO 5 STEP 2
3590 FOR x=1 TO 5
3600 LET z$(x)=a$(x,c)
3610 NEXT x
3620 LET u=VAL ("("+z$(1 TO 3)+"
)"+z$(4 TO ))
3630 PRINT AT 15,6+c*2;u
3640 NEXT c
3999 RETURN
5000 REM Rutina generacion fila
5010 LET z$(1)=STR$ (1+INT (RND*
9))
5020 LET z$(2)=s$(1+INT (RND*4))
5050 IF z$(2)<>"/" THEN GO TO 51
50
5060 IF VAL z$(1)=1 THEN GO TO 5
020
5070 FOR x=VAL z$(1)-1 TO 1 STEP
 -1
5100 IF INT (x*(INT (VAL z$(1)/x
+.5))+.5)=VAL z$(1) THEN LET z$(
3)=STR$ x: GO TO 5300
5120 NEXT x
5130 GO TO 5300
5150 IF z$(2)="-" THEN LET z$(3)
=STR$ (1+INT (RND*(VAL z$(1)-1))
): GO TO 5300
5170 LET z$(3)=STR$ (1+INT (RND*
9))
5300 LET ope=INT (VAL z$(1 TO 3)
+.5)
5320 IF ope=1 THEN LET z$(4)="+"
: LET z$(5)=STR$ (1+INT (RND*8))
: GO TO 5900
5340 IF ope<=18 THEN GO TO 5500
5350 LET z$(4)="/"
5360 FOR x=9 TO 1 STEP -1
5370 IF INT (x*(INT (ope/x+.5))+
.5)=ope THEN LET z$(5)=STR$ x: G
O TO 5900
5380 NEXT x
5390 GO TO 5000
5500 IF ope<=8 THEN GO TO 5700
5520 LET z$(4)="-"
5540 LET z$(5)=STR$ (ope-9+INT (
RND*(19-ope)))
5560 IF VAL z$(5)=0 THEN GO TO 5
540
5580 GO TO 5900
5700 LET z$(4)=s$(1+INT (RND*2))
5720 IF z$(4)="-" THEN LET z$(5)
=STR$ (1+INT (RND*(ope-1))): GO
TO 5900
5740 LET z$(5)=STR$ (1+INT (RND*
(9-ope)))
5900 RETURN
7000 REM Rutina gen. columna
7020 FOR x=1 TO 5
7040 LET z$(x)=a$(x,c)
7060 NEXT x
7070 LET t=INT (1+RND*4)
7080 LET z=INT (1+RND*4)
7090 FOR x=t TO t+3
7095 LET s=x: IF s>4 THEN LET s=
s-4
7100 FOR y=z TO z+3
7110 LET v=y: IF v>4 THEN LET v=
v-4
7120 LET z$(2)=s$(s)
7140 LET z$(4)=s$(v)
7150 LET u=VAL ("("+z$(1 TO 3)+"
)"+z$(4 TO ))
7155 LET u1=VAL z$(1 TO 3)
7160 IF INT (u1+.5)=u1 AND INT (
u+.5)=u AND u>=1 AND u<=9 THEN G
O TO 7300
7180 NEXT y
7200 NEXT x
7220 LET r$="i"
7240 RETURN
7300 FOR x=1 TO 5
7320 LET a$(x,c)=z$(x)
7340 NEXT x
7360 RETURN
9000 ERASE "m";1;"pasatiempo": S
AVE *"m";1;"pasatiempo" LINE 1:
VERIFY *"m";1;"pasatiempo"
9100 STOP
9900 SAVE "pasatiempo" LINE 1
```

CODIGO MAQUINA

CAPITULO X
(Continuación)

8.—Ejercicio

Para ver la utilización de las instrucciones que hemos estudiado en el presente capítulo, vamos a realizar una rutina que puede intercalarse en programas posteriores, incluso aunque estén escritos en Basic.

La primera de estas rutinas está en relación con la toma de datos. En esta función es muy normal encontrarse en circunstancias en las que no es posible admitir un número indeterminado de caracteres, sino que, por el contrario, tiene que ser un número determinado, o máximo. Así, sobre la marcha, se nos ocurre pensar en los números de teléfono, el número de signos de una quiniela, el distrito o código postal, la fecha, la hora, etc. En Basic no existe la posibilidad de realizar el control de los caracteres introducidos cuando utilizamos la instrucción INPUT. Para controlar este número de caracteres debemos recurrir a un bucle FOR...NEXT en el que utilizamos la instrucción INKEY $. Esta rutina tendría la forma:

```
 100 REM *** Bucle de limitacion
 de caracteres en un INPUT ***
 110 LET a$=""
 120 FOR x=1 TO l
 130 PAUSE 0
 140 IF INKEY$=CHR$ 13 THEN RETU
RN
 150 LET a$=a$+INKEY$
 160 NEXT x
 170 RETURN
```

El número de caracteres admitidos estará determinado por el valor de l, que aparece en la instrucción 120. Si, tras asignar un valor a esa variable, acudimos a la rutina mediante un GOSUB 100, en la variable alfanumérica a$ tendremos un texto con un número máximo de caracteres.

Pero nuestro tema no es el Basic, sino el Assembler, por eso vamos a pasar esta rutina a ese lenguaje. Para ello utilizamos la rutina que ya conocemos de ejercicios anteriores y que llamamos TECLA. De hecho, acudiremos a dicha rutina tantas veces como sea determinado y, cuando lleguemos a dicho límite, saldremos de la rutina. Para controlar el número de caracteres utilizaremos el registro B y la instrucción DJNZ que ya hemos visto. Esta rutina tendría, pues, el aspecto siguiente:

```
         10  *C-
8639     20  RMENL   LD     HL,(TABLA)
863C     30          LD     (DIRCA),HL
863F     40          LD     BC,(LONGI)
8643     50  RMEN1   LD     (PALON),BC
8647     60          CALL   TECLA
864A     70          CP     13
864C     80          JR     Z,FINRU
864E     90          LD     HL,(DIRCA)
8651    100          LD     (HL),A
8652    110          INC    HL
8653    120          LD     (DIRCA),HL
8656    130          LD     BC,(PALON)
865A    140          DJNZ   RMEN1
865C    150  FINRU   RET
865D    160  PALON   DEFW   0
865F    170  DIRCA   DEFW   0
8661    180  TECLA   LD     A,1
8663    190          CALL   #1601
8666    200  TECLN   CALL   #028E
8669    210          LD     C,0
866B    220          JR     NZ,TECLN
866D    230          CALL   #031E
8670    240          JR     NC,TECLN
8672    250          DEC    D
8673    260          LD     E,A
8674    270          CALL   #0333
8677    280          RET
8678    290  LONGI   DEFW   0
867A    300  TABLA   DEFW   0
```

Y ahora pasemos a comentar aquellos aspectos nuevos de esta rutina. En principio, la línea 10 contiene un directivo del programa Ensamblador en el que le decimos que no liste el Código Máquina. Este directivo es propio del Assembler que nosotros utilizamos.

En las siguientes instrucciones, de números 20 y 30, lo que hacemos es inicializar la dirección en donde vamos a ir colocando cada carácter.

A continuación hacemos lo mismo con la longitud del texto, o número máximo de caracteres del mensaje a recibir, en las líneas 40 y 50.

En la línea 60 hacemos la llamada correspondiente a la rutina TECLA, la cual no nos devuelve el control hasta que pulsemos una tecla, como ya vimos en ejercicios anteriores. El código ASC II de la tecla pulsada lo obtenemos en el registro Acumulador.

Con las instrucciones 70 y 80 lo que tratamos es de comprobar si dicha tecla ha sido la ENTER, con lo cual, en este caso, damos por terminada la introducción del mensaje, para lo cual bifurcamos a la instrucción FINRU.

Si no es la tecla ENTER, el carácter lo incorporamos al mensaje a través de las instrucciones 90 y 100, que lo que hacen es cargar el par HL con la dirección donde debe ir el carácter y almacenarlo desde el registro Acumulador. Para que el siguiente carácter no se ponga en el mismo sitio que el que acabamos de meter, es necesario sumar 1 a la dirección (HL) y volver a guardar la dirección. Esto se lleva a cabo en las instrucciones 110 y 120.

Ahora sólo queda comprobar si hemos llegado al límite previsto. Para ello se carga el número de caracteres que quedan por recibir en el registro B, y mediante una instrucción DJNZ se le resta uno y se compara con cero. Si no es cero se vuelve a repetir las operaciones desde la 50, denominada RMEN 1. Si llega al final, simplemente se finaliza la ejecución de la rutina.

Como es obvio, para que esta rutina funcione correctamente es necesario que en los campos LONGI y TABLA pongamos la longitud del texto y la dirección en donde lo queremos, respectivamente.

Su uso deberá comprenderse dentro de un programa un poco mayor, como el expuesto en capítulos anteriores, coordinándolo con la representación en pantalla del texto que introducimos.

Otra rutina que vamos a estudiar, con objeto de mostrar la utilización de la instrucción CPIR, es la que vamos a denominar «cambio de atributo». La función es muy sencilla de exponer: «Busca en el área de atributos un valor determinado y lo sustituye por otro dado». Como vamos a ver, también resulta muy fácil de llevar a la práctica, utilizando la instrucción CPIR.

Con el fin de hacerla lo más «reubicable» posible, es decir, que pueda funcionar independiente de cual sea la posición de memoria que ocupe, vamos a utilizar el área reservada para el buffer de impresora, el cual siempre se encuentra en el mismo sitio (dirección 23296), para pasar los datos que se necesitan. En el primer lugar del buffer pondremos el carácter a reemplazar, el cual corresponderá a un determinado «atributo», en cuyo valor está incluido el color del papel, el color de la tinta, la existencia, o no, de destello y la existencia, o no, de brillo, todo ello según la conocida fórmula:

$$128 * \text{flash} + 64 * \text{brillo} + 8 * \text{papel} + \text{tinta}$$

en donde,

flash y **brillo** tendrán valores de uno o cero según se desee, o no, destello y brillo respectivamente.

papel y **tinta** serán reemplazados por el código de color correspondiente en la paleta de colores del Spectrum (de 0 a 7).

En la segunda posición del buffer pondremos el atributo que sustituirá al primero.

La rutina podría tener esta forma en Assembler.

(Continúa en pág. 12)

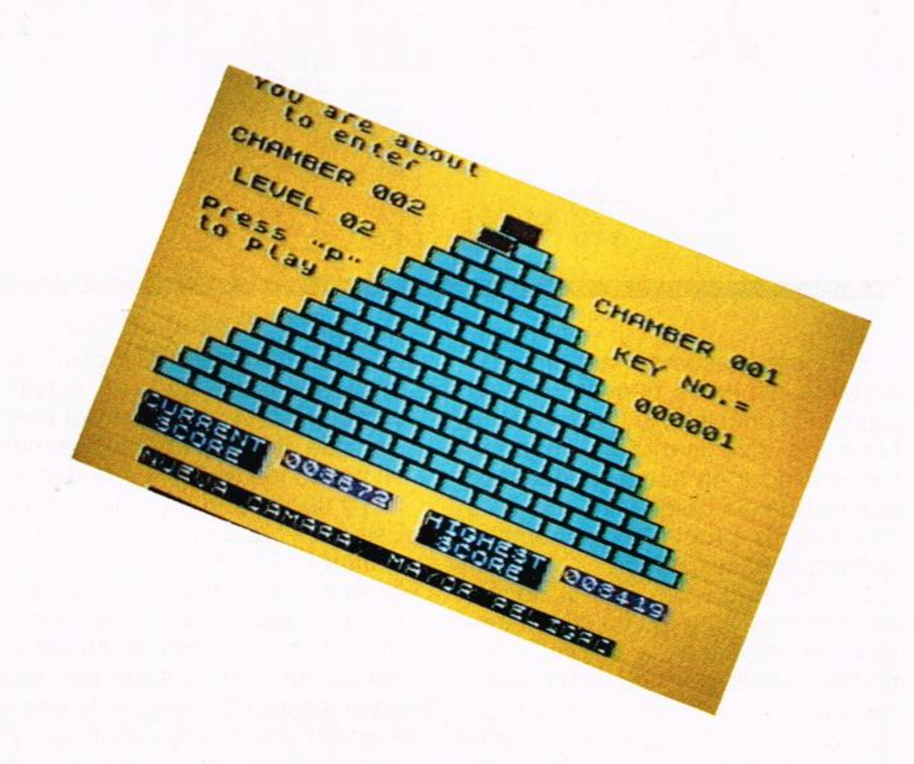
```
         10  *C-
81AA     20  CAMBA   LD     HL,22528
81AD     30          LD     BC,768
81B0     40  CAMB1   LD     A,(23296)
81B3     50          CPIR
81B5     60          JP     PO,CAMB2
81B8     70          DEC    HL
81B9     80          LD     A,(23297)
81BC     90          LD     (HL),A
81BD    100          INC    HL
81BE    110          JR     CAMB1
81C0    120  CAMB2   RET
```

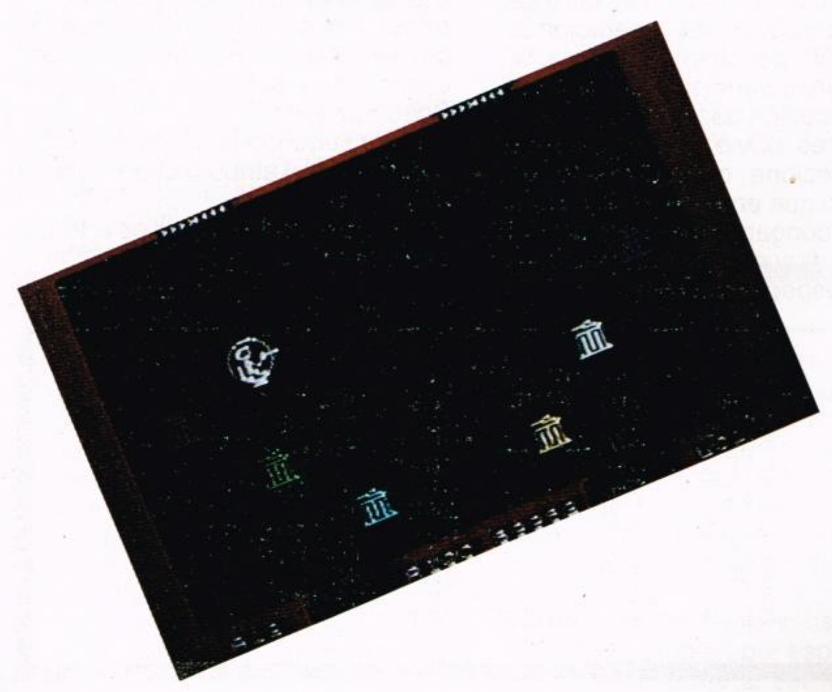
JUEGO DEL MES

Cámara secreta

En el antiguo Egipto, en lo más alto de la supremacía faraónica, Ramsés II mandó construir una gran pirámide formada por un sinfín de cámaras, en las que se supone debería perderse cualquier intruso que intentara localizar el secreto de tan temido faraón.

Cuenta la leyenda, que en el interior de cada una de las cámaras de esta gran pirámide, no sólo te asediarán terribles peligros, sino que también te encontrarás con una atmósfera viciada por el paso de los siglos y envenenada por el líquido que los sacerdotes de la época rociaron en todas las paredes de cada estancia. De cualquier modo, dispones para todo tu recorrido de





una burbuja de aire en cuyo interior te será mucho más fácil respirar y desplazarte, pues con los simples movimientos de tu joystick o con la suave presión de tus dedos, dependiendo de la opción que elijas, podrás moverte en cualquier dirección y disparar contra todo objeto que se interponga en su camino.

Dispones de un gran número de vidas y de suficiente tiempo como para conseguir alcanzar la última cámara.

Para pasar de una cámara a otra, será necesario que recojas cierto número de los diamantes que caerán por la pantalla, pero no en todos los casos será tan fácil como en la primera de ellas en la que sólo con que tomes dos de ellos y te colo-



Para empezar

Para empezar os diremos a título orientativo, que en el menú inicial, donde nos ofrece el sistema de control que deseemos utilizar, si no dispones de joystick, lo mejor será que elijas la opción que nos permite usar las teclas «Q», «A», «O» y «P», por ser las que mejor se adaptan en cuanto a relación «posición-dirección».

Para cargar

Teclear LOAD"", pulsar ENTER y poner en marcha el cassette.

¡¡BUENA SUERTE, VALIENTES!!

ques sobre la salida que te interese de la parte inferior, ésta se abrirá y te permitirá el paso, sino que a medida que avances en el juego, la dificultad aumentará, dejando a tu maravilloso intelecto, la forma de pasar de un nivel a otro, lo que dota a este ingenioso juego de un nuevo aliciente.

En el primer nivel, unos procelosos cubos de basura intentarán por todos los medios hacerte perder el mayor número posible de vidas. En el segundo, dependiendo de la puerta que elijas te asaltarán, o bien unas asquerosas mosas tse-tse, o unas rarísimas estrellas a modo de terribles arácnidos, y así sucesivamente, incrementando cada vez más la dificultad del juego.



(Viene de la página, 9)

El funcionamiento de esta rutina es igualmente sencillo. En principio se carga la dirección de comienzo del área de atributos en el registro par HL, y la longitud de dicha área en el registro par BC. El byte que queremos buscar, para sustituirlo, se carga en el registro Acumulador. Con ello cumplimos las condiciones que marca la instrucción CPIR. Esta instrucción se ejecuta hasta que se cumple alguna de estas condiciones: encuentra el byte con el valor buscado, o finaliza el área de atributos.

Lo primero que debemos averiguar, tras la cesión de control por parte de la CPIR, es el motivo por el que ha finalizado la ejecución. Para eso realizamos la instrucción JP, pues en el caso de terminarse el área tendremos que retornar al punto de llamada mediante la RET.

Si encuentra el byte buscado, tendremos en cuenta que el registro HL está apuntando al siguiente, por lo tanto, restaremos una unidad a este registro, cargaremos en el Acumulador el valor que sustituye al original, colocado en la dirección 23297, y tras almacenarlo en la posición marcada por HL, se incrementa el valor del HL para continuar de nuevo con la búsqueda. Dado que al hacer el movimiento hemos destruido el antiguo contenido del Acumulador, debemos restaurarlo mediante la carga del antiguo en el Acumulador otra vez.

La única instrucción que impide que esta rutina sea reubicable es la JP por lo que será preferible sustituirla por las siguientes instrucciones:

```
LD  A,Ø
CP  C
JR  Z, CAMB 2
CP  B
JR  Z, CAMB 2
```

Ocupan un poco más de memoria pero tienen una indudable ventaja respecto a la anterior. Quedaría así:

```
          10 *C-
81D1      20 CAMBA   LD     HL,22528
81D4      30         LD     BC,768
81D7      40 CAMB1   LD     A,(23296)
81DA      50         CPIR
81DC      60         LD     A,0
81DE      70         CP     C
81DF      80         JR     Z,CAMB2
81E1      90         CP     B
81E2     100         JR     Z,CAMB2
81E4     110         DEC    HL
81E5     120         LD     A,(23297)
81E8     130         LD     (HL),A
81E9     140         INC    HL
81EA     150         JR     CAMB1
81EC     160 CAMB2   RET
```

CAPITULO XI

1.—INTRODUCCION

Después de haber visto en el capítulo anterior las instrucciones de búsqueda por bloques, consideramos una continuación lógica de aquéllas el estudio de las instrucciones de movimientos de bloques.

Las instrucciones que vamos a tratar en este capítulo no tienen analogías con instrucciones del lenguaje Basic. Para realizar las mismas funciones que las provistas por las instrucciones Assembler deberíamos recurrir, como en muchos otros casos, a bucles FOR...NEXT, y utilizando instrucciones POKE y funciones PEEK. Por supuesto, esta fórmula implica unos tiempos extraordinariamente grandes cuando los comparemos con los que se obtienen con el código máquina.

Pero dejemos de hablar en teoría y pasemos a lo práctico.

2.—Mover con incremento (LDI)

Esta instrucción mueve un byte desde la dirección señalada por el contenido del registro par HL al lugar señalado por el contenido del par DE, incrementando el contenido de los dos registros pares en una unidad y decrementando, también en una unidad, el contenido del registro par BC.

En esta instrucción solamente se modifica el contenido del flag, o indicador, P/V que es puesto a cero si el contenido del par BC es cero, y a uno en caso contrario.

El formato de esta instrucción en Assembler es simplemente LDI en el campo de operación. No tiene operandos explícitos, pues HL, DE y BC son operandos implícitos. Su traducción a C/M es #EDAO, ocupando por lo tanto, dos bytes.

La ejecución de esta instrucción podíamos expresarla en los siguientes pasos secuenciales:

1.° Toma el byte que señala la dirección del par HL y lo almacena en la dirección señalada por el contenido del par DE.

2.° Incrementa en una unidad los registros pares HL y DE.
3.° Decrementa una unidad al registro par BC.
4.° Si el par BC es igual a cero pone el indicador P/V a cero, y si no es igual a cero, pone el indicador P/V a uno.

Tras la ejecución de este cuarto paso se da por finalizada la instrucción. Queda a nuestra voluntad el que continuemos moviendo más bytes, o no, con tal de que consultemos al indicador P/V, mediante la bifurcación incondicional oportuna.

Una rutina típica de utilización de esta instrucción sería:

```
LD   HL, Origen
LD   DE, Destino
LD   BC, Longitud
LDI
```

3.—Mover un bloque de datos (LDIR)

Esta instrucción permite mover un número de bytes determinado, contenido en el par BC, desde la dirección marcada por el registro par HL, a la dirección marcada por el registro par DE.

La ejecución de esta instrucción se realiza en los siguientes pasos:

a) Toma el byte sañalado por el contenido del par HL y lo almacena en la dirección marcada por el contenido del par DE.
b) Incrementa en una unidad los contenidos de los registros pares HL y DE.
c) Decrementa en una unidad el contenido del registro par BC.
d) Si el par BC es distinto de cero continúa la ejecución en el paso a. Si BC es igual a cero, termina la instrucción y cede el control a la siguiente.

Tras la ejecución de la instrucción, los pares de registros HL y DE estarán apuntando al final de los campos que llamaremos «destino» y «origen» respectivamente.

A diferencia de la instrucción anterior, en este caso no es necesario saber que se ha activado, o no, el indicador P/V para saber que se ha movido todo el bloque.

La utilización típica de esta instrucción está en conjunción con otras que se encargan de cumplimentar en los registros pares los valores requeridos para la correcta utilización de la instrucción. Este sería el bloque de instrucciones:

```
LD    HL, Origen
LD    DE, Destino
LD    BC, Longitud
LDIR
```

Esta instrucción, que también podríamos denominar «Mover con Incremento y Repetición», tiene su traducción a Código Máquina que es # EDBO y por lo tanto ocupa dos bytes.

4.—Mover con decremento (LDD)

Esta instrucción es similar a la LDI que hemos visto anteriormente, pero en este caso en lugar de incrementar los registros pares HL y DE los decrementa, por lo que va avanzando desde las posiciones más altas de la memoria a las más bajas.

Del mismo modo que aquélla, esta instrucción mueve un byte desde la dirección marcada por el contenido del registro par HL, a la dirección marcada por el contenido del registro par DE, decrementando a continuación el contenido de los pares HL, DE y BC.

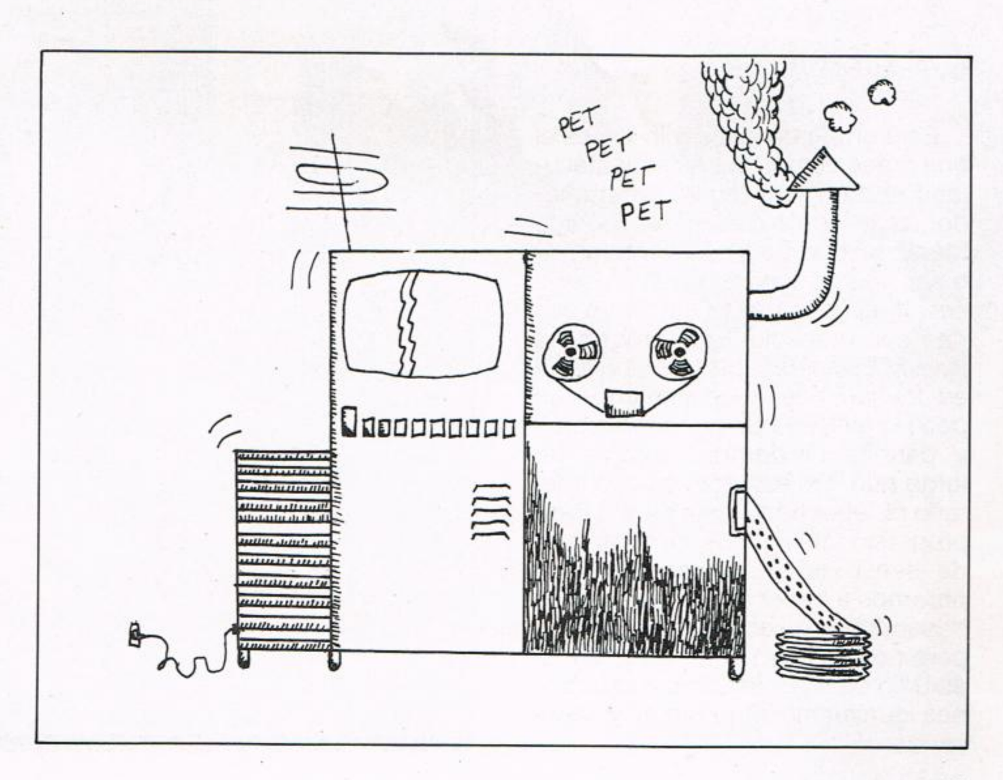
El código hexadecimal, o código máquina, de esta instrucción es ≠ED A8 y la ocupación de memoria es de dos bytes.

El único indicador que se modifica es el P/V que se pone a cero cuando el contenido del par BC es también cero, y permanece a uno en caso contrario. La forma de verificar esta circunstancia es utilizando la instrucción JP PO, dirección que solamente bifurcará a «dirección» cuando el registro par BC sea igual a cero.

Dado que ya hemos visto la secuencia de ejecución de las instrucciones LDI y LDIR no consideramos conveniente volver a insistir sobre dicho aspecto para no cansar al lector-alumno.

5.—Mover con decremento y repetición (LDDR)

Esta instrucción es la complementaria a la LDIR, del mismo modo que la LDD es la complementaria de la LDI. Es decir, se trata en este caso de mover un bloque de memoria desde una posición de memoria a otra, pero con un movimiento de «atrás hacia adelante», considerando «atrás» a las posiciones de



memoria más altas y «delante» a las posiciones más bajas de memoria.

Del mismo modo que aquélla, se van moviendo byte a byte desde la posición marcada por el contenido del registro HL, a la dirección marcada por el contenido del registro DE, decrementando los valores de los pares HL y DE, tras realizar tal movimiento, y decrementando asimismo el valor del registro BC. Dicha sentencia de proceso se repite hasta que el contenido del par BC sea igual a cero.

El código en hexadecimal, o código máquina, correspondiente a esta instrucción es #EDB8, y por lo tanto, su longitud es de dos bytes.

El control del programa no pasa a la siguiente instrucción hasta que el bloque haya sido movido totalmente, por lo que no es necesario consultar a ningún indicador para saber lo que ha sucedido.

La utilización típica de esta instrucción está en conjunción con aquellas que se encargan de cumplimentar en los registros pares aquellos valores que son requeridos para la correcta utilización de ésta. Este sería el bloque de instrucciones:

```
LD    HL, Origen
LD    DE, Destino
LD    BC, Longitud
LDDR
```

6.—Ejercicios

Este grupo de instrucciones es el que presenta una mayor espectacularidad para todo aquel programador que se inicia en el Código Máquina, porque en la utilización de estas instrucciones es quizá donde se muestra la ganancia de velocidad con respecto a los programas Basic. Este aumento de velocidad es lo que nos hace sentirnos un poco orgullosos de nuestros logros y permita olvidarnos, durante un largo rato, del esfuerzo que ha costado el llegar hasta este punto. Para proporcionar algunos motivos más de sentirnos orgullosos vamos a pararnos a hacer algunos ejercicios basándonos en estas instrucciones, pero no olvidaremos el intercalar el estudio de algunas otras instrucciones igualmente importantes y necesarias.

Para comenzar, «abriremos boca» con una sencilla rutina que nos permitirá borrar la pantalla, como cuando utilizábamos la CLS en Basic. Ya sabemos, porque en ejercicios anteriores la hemos utilizado, que existe una rutina en la ROM que ya realiza esta función, pero dejando aparte el hecho de que esa rutina, concretamente situada en la posición #OD6B de la memoria, además de borrar, inicializa las posiciones de línea y columna de la próxima PRINT, y otras cosas más, nuestro objetivo no ha sido nunca el enseñar a manejar las rutinas de la ROM, sino el lenguaje Assembler, y el Código Máquina, por lo que nos crearemos nuestra propia rutina de borrar pantalla.

Para hacerlo realidad bastará poner toda el área de imagen a ceros binarios. Dado que estas instruccciones permiten mover un byte direccionado por el par HL a la dirección del par DE, bastará escoger correctamete el lugar que deberán direccionar ambos registros pares para que un cero binario se propague por toda el área con la utilización de una única instrucción, la LDIR.

Veamos ahora, de modo práctico, la puesta en práctica de la teoría expuesta en el párrafo anterior. La rutina tendrá esta forma:

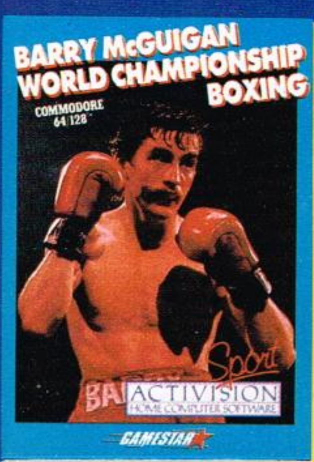
En primer lugar lo que hacemos es poner a cero el registro Acumulador y almacenarlo en la primera posición del área de pantalla. A continuación cargamos en el registro par HL la dirección de ese byte y en el par DE la del siguiente. La longitud del área de pantalla, incluidos los atributos, es de 6912 bytes, y esa es la cantidad que movemos al par BC, pero como el primer byte de dicha área ya está puesta a cero, le restaremos una unidad a esa cantidad y sólo pondremos 6911. Sólo nos queda ejecutar la instrucción LDIR para que todo el área esté a ceros binarios.

Para comprenderlo, tal vez sea conveniente recordar la ejecución de la instrucción LDIR con un poco de atención. Este uso de la LDIR es un poco complicado de comprender si pensamos que la instrucción trabaja moviendo bloques, olvidándonos de un pequeño matiz: el bloque lo mueve byte a byte. De este modo, cuando va a mover el segundo byte resulta que éste ya contiene la misma información que el anterior y en eso consiste el truco.

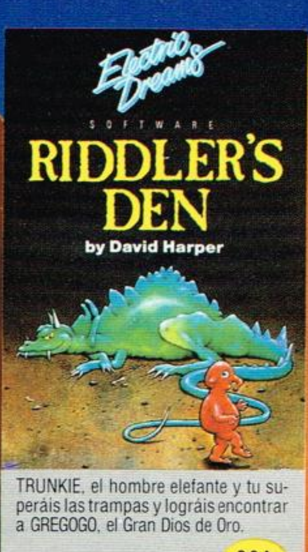
Junto al listado digamos «educativo» de la rutina, ponemos el listado de la rutina «real». Con ello queremos que el lector tenga una visión real de la programación, y de cómo llegar a ella.

```
          10 *C-
817C      20 RUCLS   LD    A,0
817E      30         LD    (16384),A
8181      40         LD    HL,(16384)
8184      50         LD    DE,(16385)
8188      60         LD    BC,6912
818B      70         DEC   BC
818C      80         LDIR
818E      90         RET
```

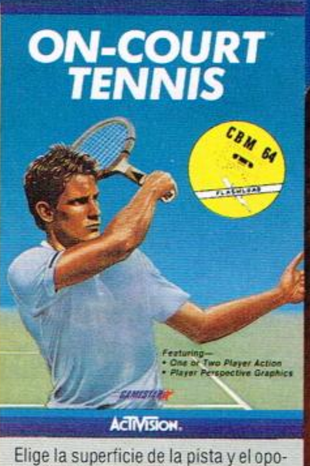
```
          10 *C-
816F      20 RUCLS   LD    A,0
8171      30         LD    HL,(16384)
8174      40         LD    (HL),A
8175      50         LD    DE,(16385)
8179      60         LD    BC,6911
817C      70         LDIR
817E      80         RET
```



Juego en el que puedes crear a tu propio boxeador. Elige su raza, estilo físico e imagen. Entrénale y demuestra sus habilidades. C.S.A.



TRUNKIE, el hombre elefante y tu superáis las trampas y lográis encontrar a GREGOGO, el Gran Dios de Oro. C.S.A.



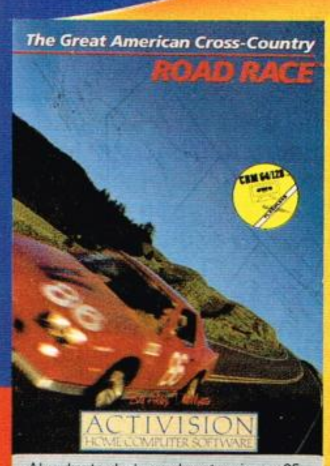
Elige la superficie de la pista y el oponente y demuestra tu control de la raqueta jugando a dobles o individual. C.S.A.



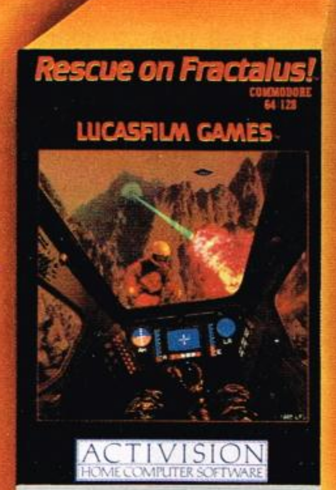
Entra en el mundo ciclista a través de las 16 etapas del Tour. Con acompañamiento musical y el jersey amarillo esperando al ganador. C.

ACTIVISION INC
HOME COMPUTER SOFTWARE

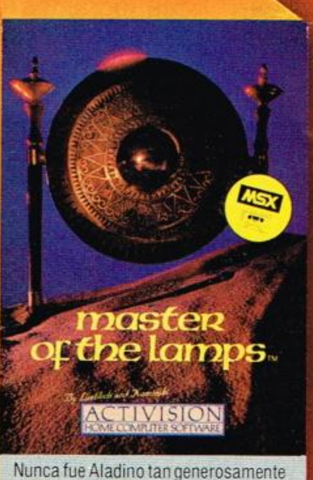
*P.V.P 2.200



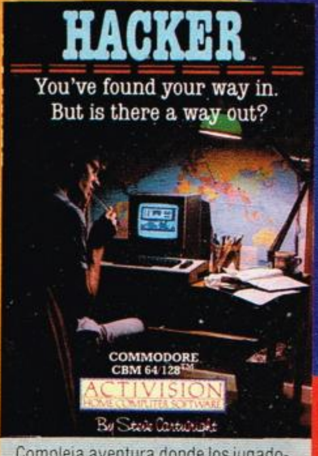
Al volante de tu coche atraviesas 25 ciudades. Seleccionas la ruta, maniobras a través del tráfico... Todo un reto de conducción automovilística. C.A.



Recorre un planeta en tres dimensiones a la búsqueda de tus compañeros, mientras el enemigo te persigue. C.S.A.



Nunca fue Aladino tan generosamente premiado por los genios.
Vuele sobre una increíble alfombra mágica en tres dimensiones. C.A.M.



Compleja aventura donde los jugadores deben buscar a través de las diferentes pistas y problemas como resolver el misterio. C.S.A.



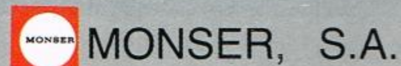

... y sus clásicos: GHOSTBUSTER. C.S.A.M. DECATHLON DE ACTIVISION. C.M. RIVER RAID. C.S.M.

* ENDURO S PITFALL 2 C.S.M. y otros títulos más a 1.540.- P.V.P. • SPACE SHUTTLE. C. DESIGNER PENCIL. C. ... etc.

Disponibles para:
COMMODORE C
SPECTRUM S
AMSTRAD A
MSX M

EN TIENDAS ESPECIALIZADAS Y GRANDES ALMACENES
O DIRECTAMENTE
POR CORREO O TELEFONO A:

Argos, 9 – 28037 MADRID
Teléfs. 91/742 72 12/96



ANALISIS

NIGHTSHADE

Mis ojos se llenan de lágrimas, tiemblan mis manos al depositar la cabeza de mi padre sobre la almohada. Con una intensa emoción cierro sus ojos; todavía resuenan en mis oídos sus últimas palabras: «¡Olaf, hijo mío, salva a Nightshade, libra de las tinieblas a tu pueblo!»

Una avalancha de recuerdos afluyen a mi memoria, parece como si estuviera viviéndolo, recuerdo cuando en mi pueblo, Nightshade, vivíamos felices, los niños jugaban en las calles, los viejos discutían y cotilleaban sentados en la gran piedra de la plaza grande, los hombres labraban contentos, mientras miles de voces femeninas se escuchaban cantando felices realizando las tareas de la casa.

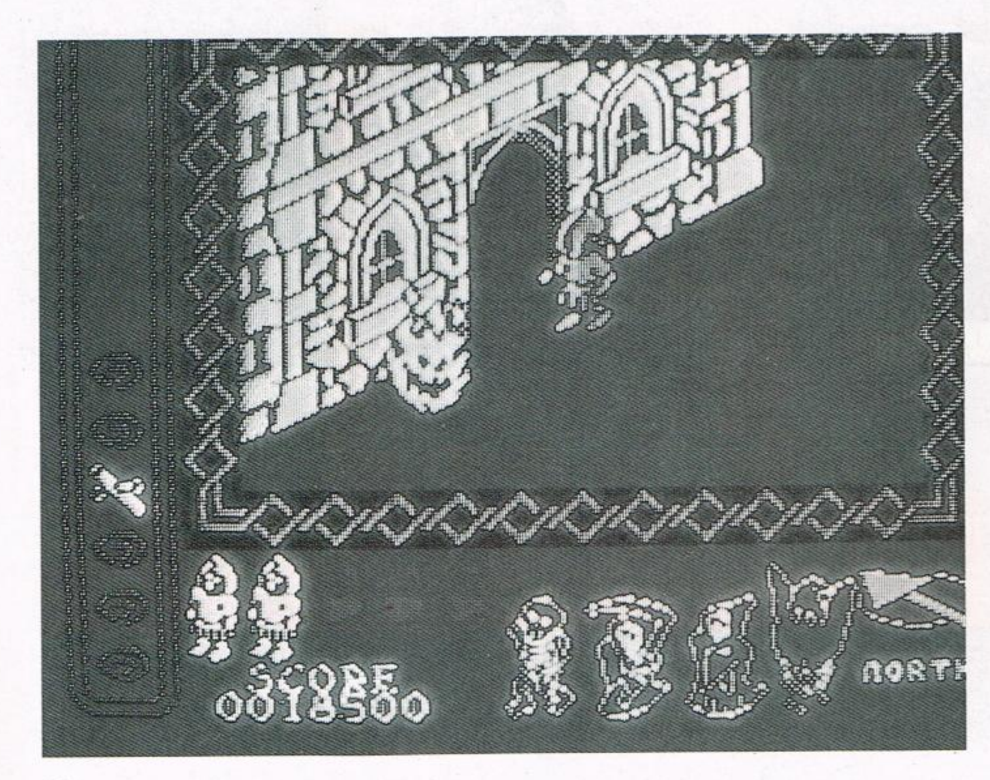
Recuerdo el infortunado día en que, como una plaga, llegaron los cuatro seres enviados por Satanás para adueñarse del pueblo. Todavía tiemblo de pánico cuando recuerdo su llegada, la luz se disipó, todo se cubrió de oscuridad, empezaron a convertir en esclavos a mis paisanos, sólo unos pocos habitantes logramos escondernos en una bodega, y así pudimos librarnos de la esclavitud, pero éramos conscientes de lo circunstancial de la situación, por eso votamos para que uno de nosotros saliese para intentar eliminar a los señores del mal.

Recuerdo todavía la sensación, mezcla de temor y orgullo, cuando salió elegido mi padre, el guerrero más sabio del pueblo.

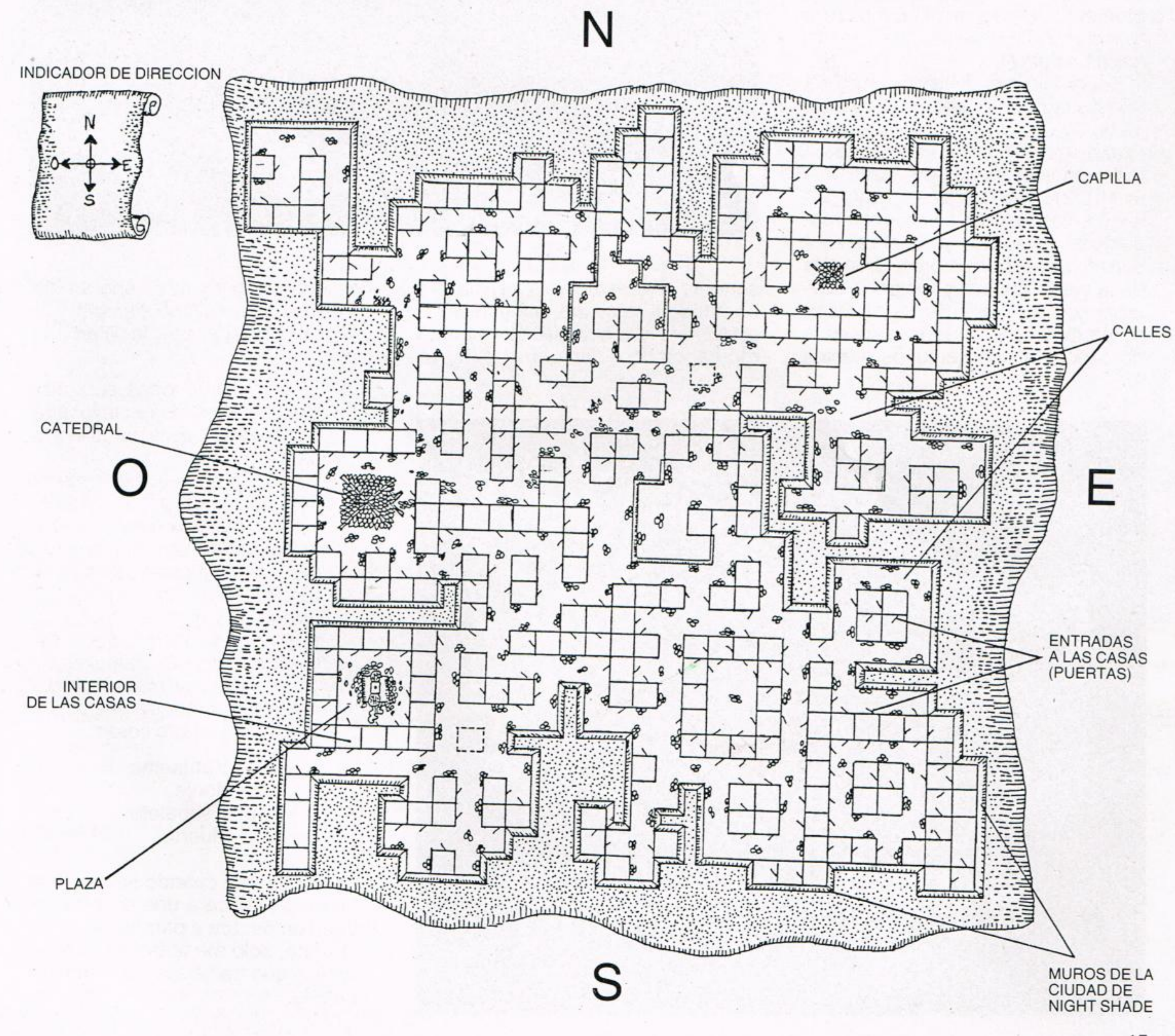
Recuerdo que estuvo cerca de un mes fuera de casa, cómo sufríamos al no tener noticia de él, pero sabíamos que era prudente tal situación, ya que si no podríamos ser descubiertos.

Recuerdo la horrible sensación que sentí cuando un infortunado día mi padre volvió moribundo. Nos contó todas las batallas y escaramuzas que había tenido con los señores del mal y las tinieblas. No puedo olvidar cómo lloraba de tristeza cuando exhaló el último hálito de vida y murió.

Estoy de rodillas ante su cadáver, sus manos en las mías, una terrible sensación de desamparo se va apoderando de mí, pero... ¡No!, ésto no lo querría él, lucharé yo contra los que nos sojuzguen, juro que vengaré tu muerte, padre. Temblando de ira me retiro a un rincón y procuro calmarme, intento poner en orden mis ideas y procuro esbozar un plan de ataque, he de recordar bien la narración de mi padre, para obtener los remedios con los que combatirlos. Recuerdo que habló de unos objetos que traía en su macuto... ¡Ah, ya! eran cuatro, un reloj de arena con el que se combate la muerte, un mazo contra el esqueleto, una biblia contra el fantasma y

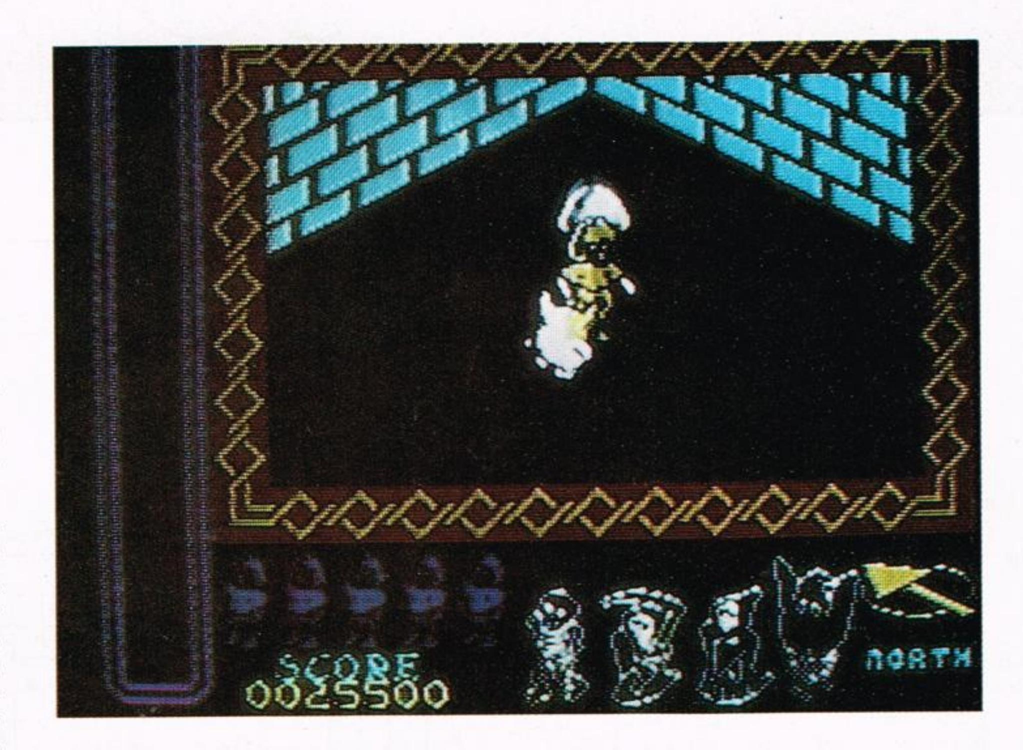


SOFTWARE


N
INDICADOR DE DIRECCION
N
O
E
S
CAPILLA
CALLES
CATEDRAL
O
E
ENTRADAS
A LAS CASAS
(PUERTAS)
INTERIOR
DE LAS CASAS
PLAZA
S
MUROS DE LA
CIUDAD DE
NIGHT SHADE

una cruz contra el monje. Cojo el macuto y observo su interior, veo los cuatro objetos esperándome, veo también armas y recuerdo que mi padre habló de unos siervos de los señores del mal a los que se les puede matar con ellos, los señores son inmunes a ellas. Cierro el libro y sin decir nada a mis abatidos compañeros de clandestinidad, salgo por la puerta de la bodega en silencio, siento el frío en el rostro, la oscuridad me rodea, por un momento se encoje mi ánimo, pero reacciono al momento, ¡He de vengar a mi padre!

Bueno amigos, esta es la historia en la que puede estar basado este juego de *Ultimate*, la casa que logra rizar el rizo en cuanto a gráficos tridimensionales, varía en cuanto a NIGHTLORE, en que este juego son los objetos los que se mueven respecto al personaje y no al revés y son utilizados tres colores. Creo que es la característica principal a destacar en este juego. Porque la idea no es muy original, la música es la de costumbre con algunos efectos sonoros interesantes. Creemos, y en nuestra humildad, aconsejamos a *Ultimate* que cambie, que ya está muy agotado el filón que comenzó con SABRE WULF y que tiene a este programa como último ejemplo.



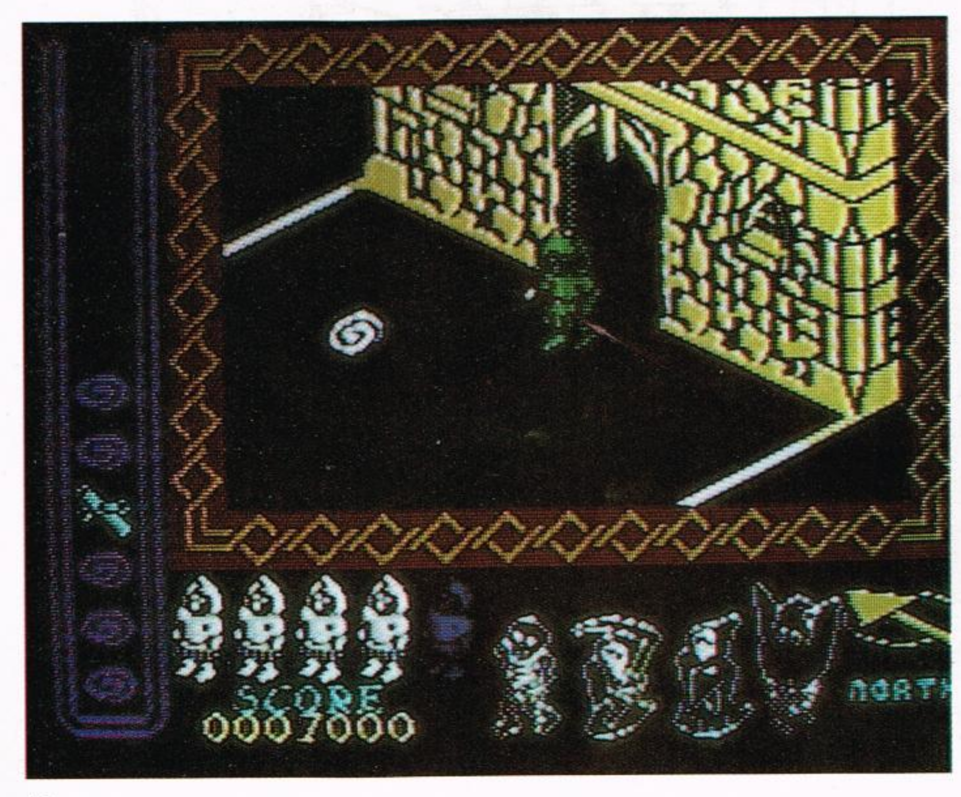
Unos consejos para facilitaros la tarea:

a) El número de vidas se señaliza con unos muñecos (5) que están debajo de la pantalla a la izquierda.
b) La energía de nuestro héroe se va notando por su progresivo empalidecimiento, puede aumentarla con una pócima que hay en alguna pantalla.
c) Los monstruos que nos atacan pueden ser destruidos con las armas que encontramos repartidas entre las pantallas.
d) Los señores del mal son inmunes a todo menos a una de las siguientes cuatro cosas:

Biblia - Fantasma
Cruz - Monje
Mazo - Esqueleto
Reloj - Muerte

Notaréis que cuando llevamos el objeto que ataca a uno de ellos, el objeto empezará a parpadear.

Bueno, sólo me falta desearos el triunfo y que traigáis la luz a vuestra pantalla.



SOFTWARE + LIBRO DE TEXTO + GUIA DE TESTS

Cursos Educativos

Programa informatico de estudio y repaso para estudiantes de E.G.B. (últimos cursos), B.U.P. y C.O.U.



En los CURSOS EDUCATIVOS MONSER aparecen combinados de modo sumamente original el libro y el ordenador, en ellos el microordenador desempeña la misma función que un profesor particular.

Cada curso comprende:

— Un magnífico libro de texto: Es también manual de repaso y proporciona explicaciones, orientaciones y preguntas prácticas.

— Cassette 1. Módulos de aprendizaje: Se aprovechan las posibilidades interactivas del ordenador, para facilitar al alumno las partes más importantes del programa de estudios, con ejemplos muy elaborados y preguntas cuidadosamente estructuradas; todo con ayuda de diagramas dinámicos.

— Cassette 2. Exámenes prácticos: Evalúa hasta qué punto se conoce la asignatura, indica los temas comprendidos y los que están flojos, familiariza con los exámenes reales, proporciona un plan de estudios y repaso a la medida de las necesidades del alumno.

— Una guía para el estudiante: Contiene todos los test de diagnósticò e intrucciones para operar. Es una ayuda inestimable para sacar el máximo provecho del curso.

Cualquier estudiante que se ocupe intensamente de este curso comprobará los resultados a la hora de examinarse.

MONSER

- Deseo que me envien los cursos marcados con una X al precio de 3.700 ptas. (IVA incl.) cada uno*

	Spectrum	Commodore	Amstrad	Msx
MATEMATICAS	☐	☐	☐	☐
QUIMICA	☐	☐	☐	☐
BIOLOGIA	☐	☐	☐	☐
FISICA	☐	☐	☐	☐

TOTAL .... Ptas.

Forma de pago:

☐ Cheque adjunto ☐ Giro Postal N .º.............. ☐ Contra reembolso

NOMBRE ....

DIRECCION ....

POBLACION .... C.P. ....

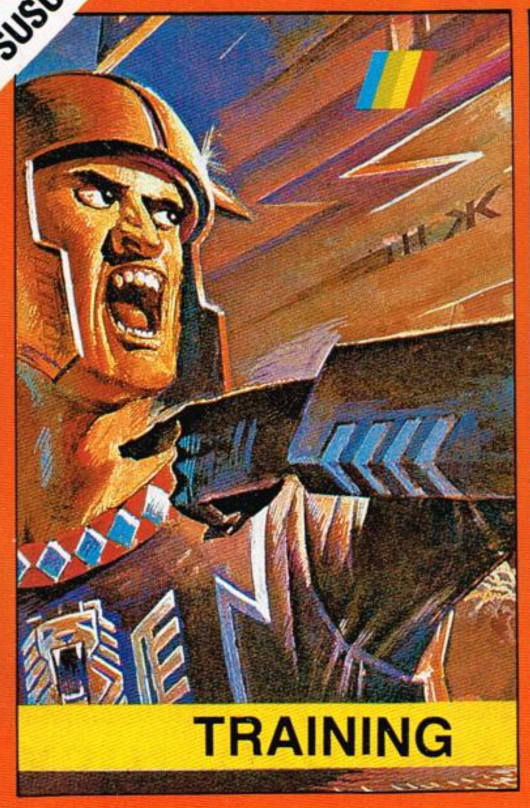
PROVINCIA .... TEL.: ....

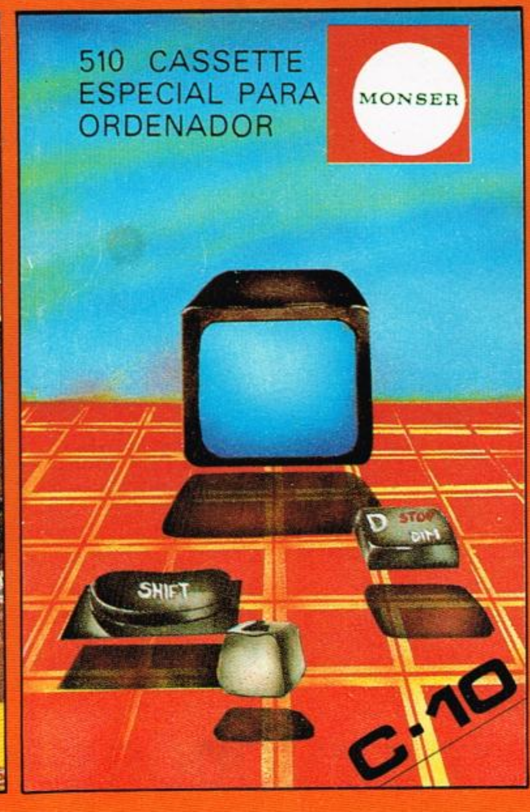
FECHA .... FIRMA:

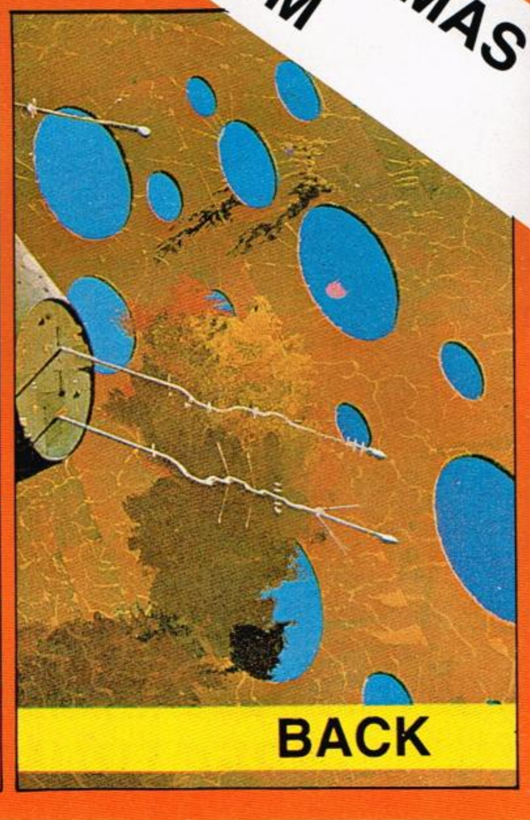
* Los gastos de envio han de ser abonados a la entrega del pedido.

* Proximamente

Disponible para: SPECTRUM *COMMODORE *AMSTRAD *MSX


TRAINING


510 CASSETTE
ESPECIAL PARA
ORDENADOR
MONSER
SHIFT
C-10


BACK